# 动态建模

**概述：**本次动态建模工作主要聚焦于“采摘任务分配”、“低电量应急分配”和“任务状态机”三个核心方面。建模成果包括两个时序图和一个状态图，均配有相应的规约描述，并说明了其与类对象模型的对应关系。

（所有图片清晰度不够，附件有原图。）

## 时序图：

### SD-001 分配采摘任务（UC-001，正常+扩展）

#### 规约描述

- **目的：**接收环境/识别结果，生成任务并依据车辆能力与约束完成优先级计算、任务分配与指令下发。
- **参与对象：**dc:DecisionController，gw:OpsGateway，ws:WorldState，mm:MissionManager，ae:AssignmentEngine，sg:SafetyGuard，t*:Task。
- **触发/前置：**控制循环 dc.tick() 或操作员触发；UAV/UGV 在线并可通信；世界状态可刷新。
- **主成功场景（与图中消息一一对应）**
  S1. dc→gw: pullEnv()；gw→ws: refresh(world) 完成融合。
  S2. dc→mm: createTasks(ws.targets)，mm 依据目标生成 Task。
  S3. dc→ae: rankAndAssign(t*, ws.ugvs) 计算优先级与分配方案。
  S4. dc→sg: check(assignment) 校验安全/并发/能耗等约束。
  S5. dc→gw: sendUGV(assignment) 下发到目标 UGV。
  S6. dc→mm: markAssigned(t, ugv) 持久化分配结果。
- **扩展/异常（图中 loop/alt/break 已体现）**
  E1（循环）：对每个候选任务执行 S3 – S6。
  E2（车状态异常）：mm.exclude(ugv)，回到 S3 重算。
  E3（识别质量差）：dc→gw: requestRescan(uav)，任务降权/待重扫。
  E4（通信中断）：break 使用 lastKnown 继续评估，通信恢复后 resume 再持久化。
  E5（约束不满足）：sg 失败→回 S3 改派。
- **后置：**任务状态从 Pending/Queued→Assigned；派发记录与审计日志落库。

# 动态建模

- **性能/非功能**：分配计算 ⩽ 5s；重分配 ⩽ 3s；数据融合 ⩾ 20Hz；端到端控制延迟 ⩽ 200ms；日志完整、幂等（重复下发不造成重复执行）。

## 与类/对象建模的对应性

- **控制—服务—实体**分层：DecisionController、OpsGateway/MM/AE/SG，实体为 Task/WorldState；与类图的依赖/聚合方向一致。
- **关键属性/约束**：使用 Task.priority/status/assignedTo、UGV.capacity/battery、安全阈值（minBattery/maxConcurrent）等；与对象图中的实例名（如 dc1/mm1/ws1）和状态变化（Assigned）一致。

## 图片如下：

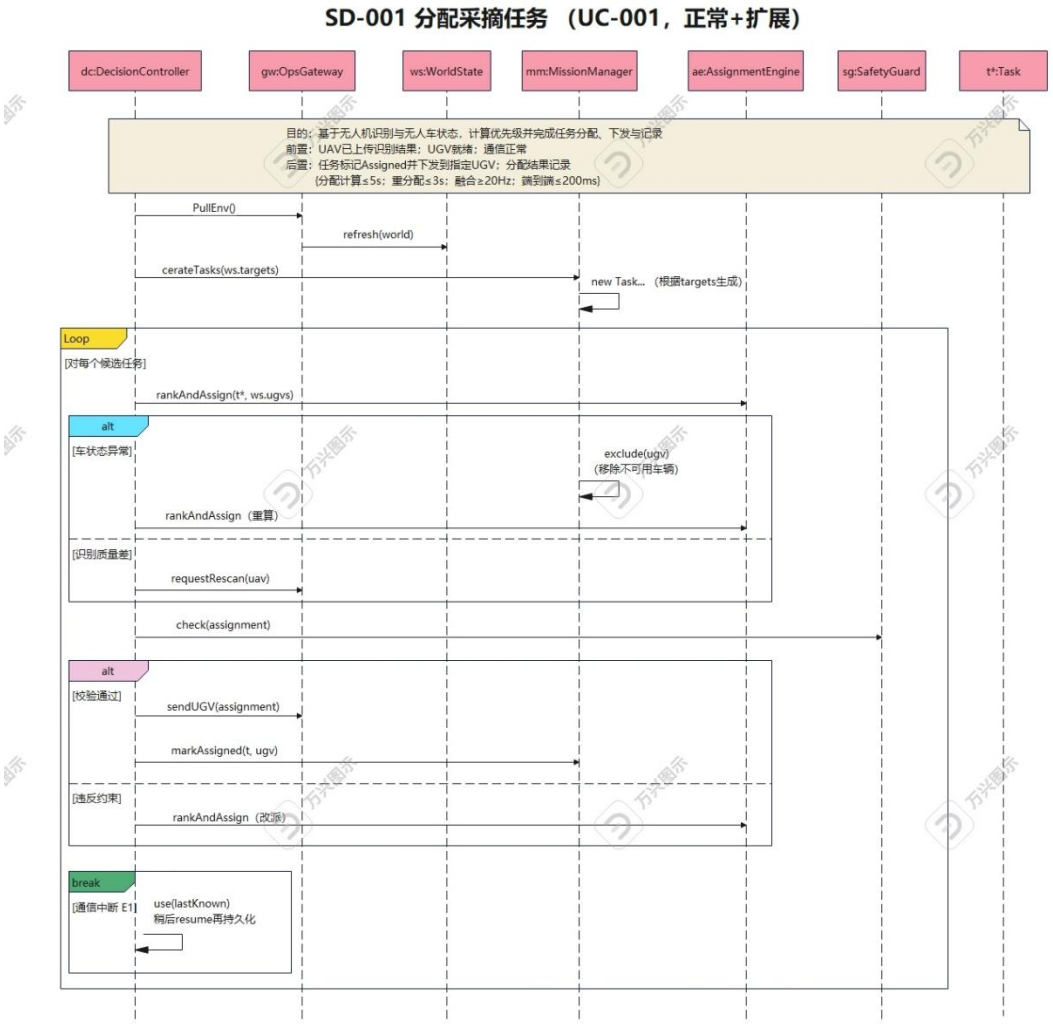


图 1 SD-001 分配采摘任务

# 动态建模

## SD-004 低电量应急重分配（UC-004，告警→恢复）

### 规约描述

- **目的**：当 UGV 低电量/异常时，制定恢复计划（返航/重派/重发）并执行，保证连贯作业与安全。
- **参与对象**：fm:FaultMonitor（或告警源）、dc:DecisionController、sg:SafetyGuard、ws:WorldState、gw:OpsGateway、mm:MissionManager、ra:RecoveryAction（数据对象）。
- **触发/前置**：fm→dc: alarm(LOW_BATTERY, ugv)；监控启用、预案配置可用、链路正常。
- **主成功场景**
  R1. dc→sg: recoverPlan(ws, task)，sg→ra: new(plan) 产出步骤（如 Return/Reassign）。
  R2. dc→gw: apply(Return(ugv)) 下发返航/安全动作。
  R3. dc→mm: reassign(task, ugv') 选择备车；dc→gw: sendUGV(Go(target) for ugv')。
  R4. dc→mm: update(task.assignedTo=ugv') 并通知操作员。
- **扩展/异常**
  A1（告警抖动）：opt debounce 5–10s 再执行。
  A2（自动处理失败）：alt 升级人工介入 escalateToHuman()。
  A3（通信异常）：安全动作优先（返航），通信恢复后继续 R3–R4。
- **后置**：恢复动作已执行；原任务被安全接管/重派；审计记录与通知完成。
- **性能/非功能**：检测响应 ≤ 1s；恢复流程 ≤ 30s；误报控制，告警准确率 ≥ 95%；关键步骤可追踪（记录 ra、责任人、时间戳）。

### 与类/对象建模的对应性

- FaultMonitor 事件驱动 DecisionController；SafetyGuard 产出 RecoveryAction；MissionManager 修改 Task.assignedTo；OpsGateway 执行设备级动作。
- 使用 UGV.battery、Task.status/assignedTo、RecoveryAction.steps 等属性；与对象图中的“返航/重派”场景一致。

### 图片如下：

# 动态建模

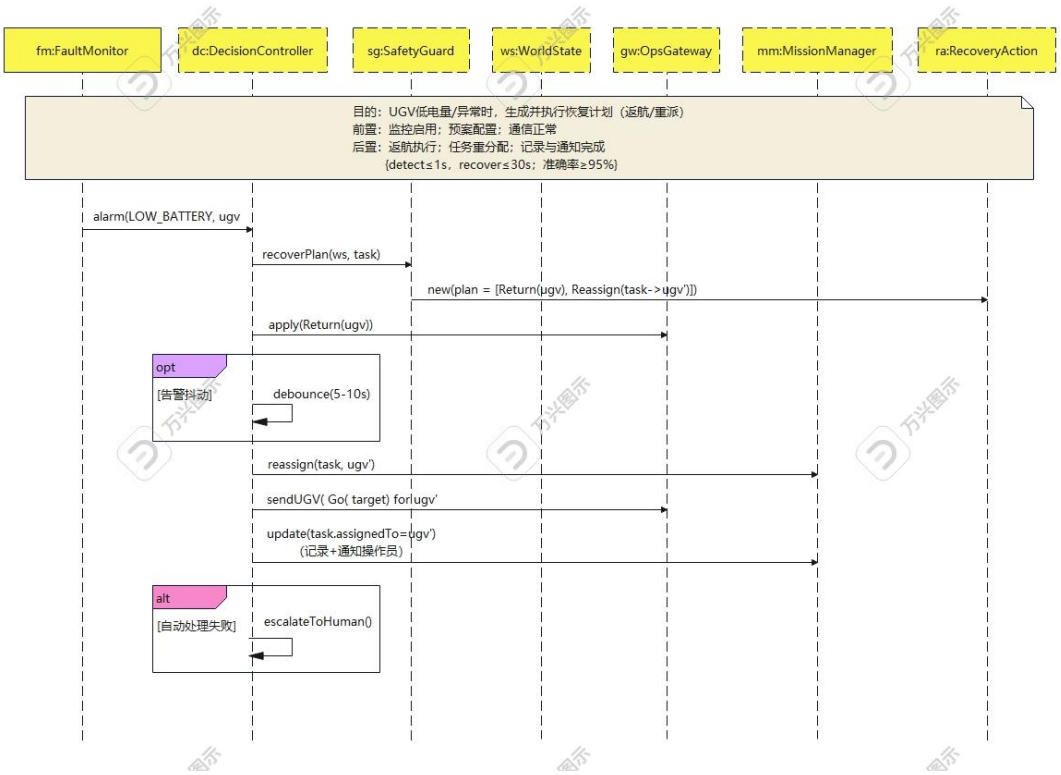


图 2 SD-004 低电量应急重分配

## 状态图：

## STM-Task 任务状态机

## 规约描述

- **建模对象**：Task（实体类）。
- **状态集合**：
  Pending（初建）→Queued（入队）→Assigned（复合：AwaitAck→AwaitReady）→Dispatched→InProgress（复合：Navigating→Working）→ 终态 Completed / Failed / Canceled；支路 OnHold、Reassigning。
- **关键事件/触发**：rankOK，assign(ugv)，ack，ready/arrived，harvestDone，batteryLow，pathUnreachable，timeout，collision，cancel。
- **守卫/动作示例**：
  - Assigned.AwaitAck→AwaitReady [ack<=1s] / logAssign()
  - Dispatched→InProgress: arrived
  - InProgress.Navigating→OnHold: pathUnreachable / requestRescan()
  - InProgress.Working→Completed: harvestDone

  - InProgress→Reassigning [battery<30%] / dc.requestRecover() → Reassigning→Assigned / mm.selectNewUGV()
  - 任意→Canceled : cancel

- **entry/exit/do：**
  Pending.entry/initContext()；Queued.entry/enqueue()；Dispatched.entry/pushRoute()；InProgress.entry/startTimers()、do/reportTelemetry()、exit/stopTimers()。
- **不变量：**Assigned ⇒ ugv.capacity ≥ task.load；InProgress ⇒ path.feasible ∧ comms.ok。
- **终结：**Completed/Failed/Canceled → [*]（确保模型有结束伪结点）。

## 与类/对象建模的对应性

- 状态字段 Task.status 与类图/对象图一致，转移由 MissionManager.assign()、OpsGateway.pushRoute()/ack、SafetyGuard.requestRescan()/recoverPlan() 等方法触发；
- 约束与属性：ackDeadline、battery、capacity、path.feasible 等与静态建模中的属性与关联一致；对象图中 Assigned/Return 的快照能在此状态机找到对应切面。

## 图片如下：

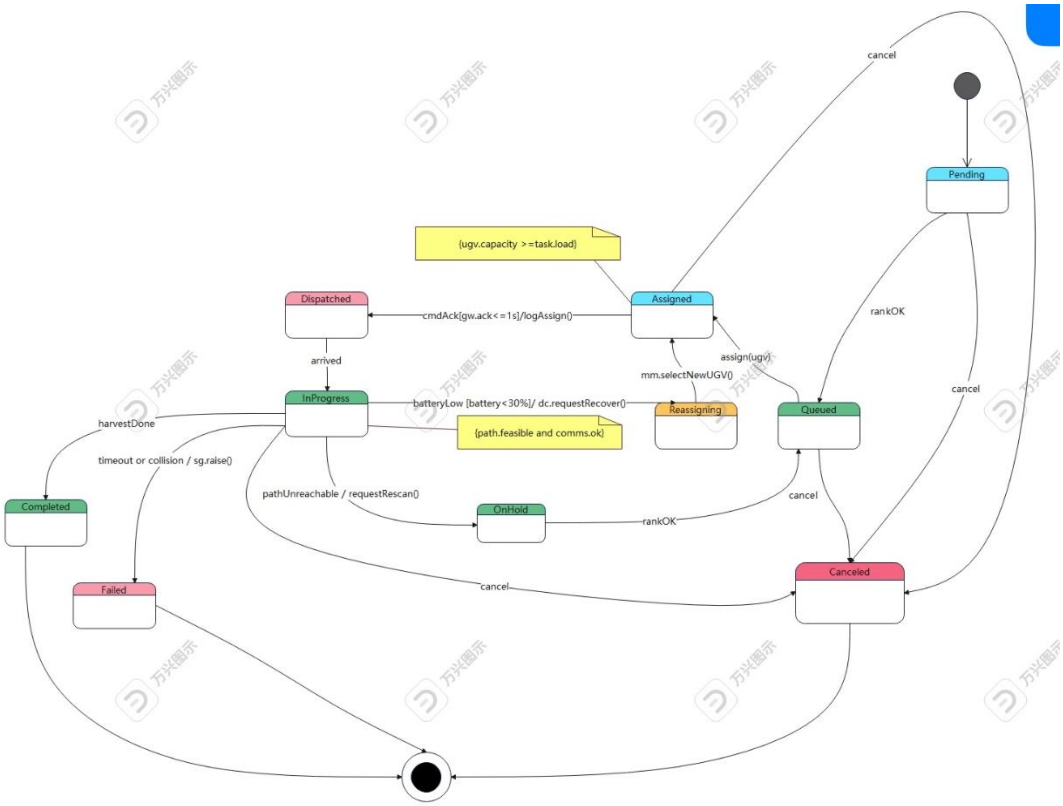


图 3 STM-Task 任务状态机

# 动态建模

## 统一说明（放在报告“总则/一致性”小节）

- **命名一致**：图中的对象名、消息名与类图/对象图完全一致（dc/mm/ws/ae/sg/gw 等；消息小驼峰）。
- **范围对齐**：两张时序图分别覆盖 UC-001、UC-004；状态机覆盖 Task 全生命周期，三者共同覆盖“生成→分配→执行→恢复/终结”的核心业务链路。
- **非功能要求**：在图注中标出时间阈值、频率、可靠性与日志要求；与需求/用例文档一致。